LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-CONF-541611

# Detection of Coherent Structures in Extreme-Scale Simulations

C. Kamath, J. Iverson, R. Kirk, G. Karypis

March 26, 2012

Exascale Research Conference
Portland, OR, United States
April 16, 2012 through April 18, 2012

## Disclaimer

# Detection of Coherent Structures in Extreme-Scale Simulations

Chandrika Kamath[1], Jeremy Iverson[2], Roger Kirk[1], and George Karypis[2]
[1] Lawrence Livermore National Laboratory, Livermore CA 94550
[2] University of Minnesota, Minneapolis MN 55455

**Abstract**

The analysis of coherent structures is a common problem in many scientific domains ranging from astrophysics to combustion, fusion, and materials science. The data from three-dimensional simulations are analyzed to detect the structures, extract statistics on them, and track them over time to gain insights into the phenomenon being modeled. This analysis is typically done off-line, using data that have been written out by the simulations. However, the move towards extreme scale architectures, with multi-core processors and graphical processing units, will affect how such analysis is done as it is unlikely that the systems will support the I/O bandwidth required for off-line analysis. Moving the analysis *in-situ* is a solution only if we know a priori what analysis will be done, as well as the algorithms used and their parameter settings. Even then, we need to ensure that this will not substantially increase the memory requirements or the data movement as the former will be limited and the latter will be expensive. In the Exa-DM project, a collaboration between Lawrence Livermore National Laboratory and University of Minnesota, we are exploring ways in which we can address the conflicting demands of coherent structure analysis of simulation data and the architecture of modern parallel systems, while enabling scientific discovery at the exascale. In this paper, we describe our work in two areas: the *in situ* implementation of an existing algorithm for coherent structure analysis and the use of graph-based techniques to efficiently compress the data.

**Keywords:** Data movement, algorithms

## 1 Introduction

The detection and analysis of coherent structures is a problem that occurs in many domains. In fusion, coherent structures in both simulation and experimental data are analyzed to understand the effects of turbulence that could result in the loss of confinement of the plasma [11, 8] In combustion, the behavior of these structures provides insights into the process of ignition, extinction, and re-ignition. In materials science, such analysis can be used to understand properties of metals through simulations that model the formation of macroscopic grains that grow out of a liquid and form structures when the liquid is subject to high temperature and pressure [14]. Coherent structure analysis also plays a role in understanding instabilities in the mixing of fluids, such as the analysis of bubble and spike structures in Rayleigh-Taylor instability [6] or the validation of the Richtmyer-Meshkov instability [7].

A typical scenario in the analysis of coherent structures involves the writing out of the data from simulations or experiments, followed by the application of various analysis algorithms off-line. The concept of a coherent structure is often an intuitive one, where a group of grid points (or pixels in an image) is considered to be a structure if the points are spatially close and behave in a similar manner. As the analysis of these coherent structures is usually motivated by scientific discovery and the need to to gain insights into the phenomenon being simulated (for example, does the distribution of the size of the structures follow a power law?), it precludes a quantitative definition that could be used to extract the structures from data. Consequently, we need to experiment to identify algorithms that are most appropriate for a given dataset, and use different algorithms to gain confidence in the results. Further, as these structures evolve with time, and may merge, split, appear, or disappear, it can be difficult to select the parameters for the algorithms used in the analysis.

Addressing all these issues is a challenge, especially when the datasets are very large, being measured in tens of terabytes or more. Exacerbating matters further is the recent move towards extreme-scale computing

using multi-core processors and graphical processing units. On one hand, this provides the compute power to make these massive-scale simulations feasible, while on the other hand, the new architectures will have severely limited I/O bandwidth, thus affecting the amount of data that can be written out. In addition, the high cost of data movement within a processor, and the limited memory, also pose challenges as we seek to enable scientific discovery in simulations at the extreme scale.

The Exa-DM project, a collaboration between Lawrence Livermore National Laboratory and the University of Minnestoa, seeks to address these challenges in different ways. In this paper, we focus on two solution approaches - the first considers ways in which an existing technique for detection of coherent structures is being modified to perform the analysis *in-situ* and the second explores the use of clustering techniques from data mining to intelligently reduce the size of the data being output.

## 2 Coherent structures in simulation data

In a problem involving coherent structure analysis of simulation output, the data consist of the values of different variables at grid points in the problem domain. The underlying grid can be structured or unstructured and the number of variables depends on the problem. Sometimes, the domain scientists might output only those variables which are of immediate interest. If the initial analysis indicates that other variables must be analyzed, then the simulation is re-run to write out additional data.

Figure 1 is a small subset of the data output from a plasma physics simulation that was run as part of the GSEP Fusion SciDAC Center [10, 3]. This center is developing the predictive capability for assessing the effects of energetic particles on the performance of the burning plasmas in ITER [4]. The figure shows two variables, electrostatic potential and ion heat flux, on a single poloidal plane. The full dataset is in three dimensions, with 32 poloidal planes spaced equidistant on a toroid. The dataset shown is relatively small, with 40000 grid points in a poloidal plane and five variables at each grid point. The analysis is done on more moderate-sized datasets, with 64 poloidal planes, each with 600,000 grid points and 10 variables per grid point, run over 8000 time steps. The different datasets are the result of modeling using different physics, with the coherent structure analysis being used to provide insights into the different approaches.

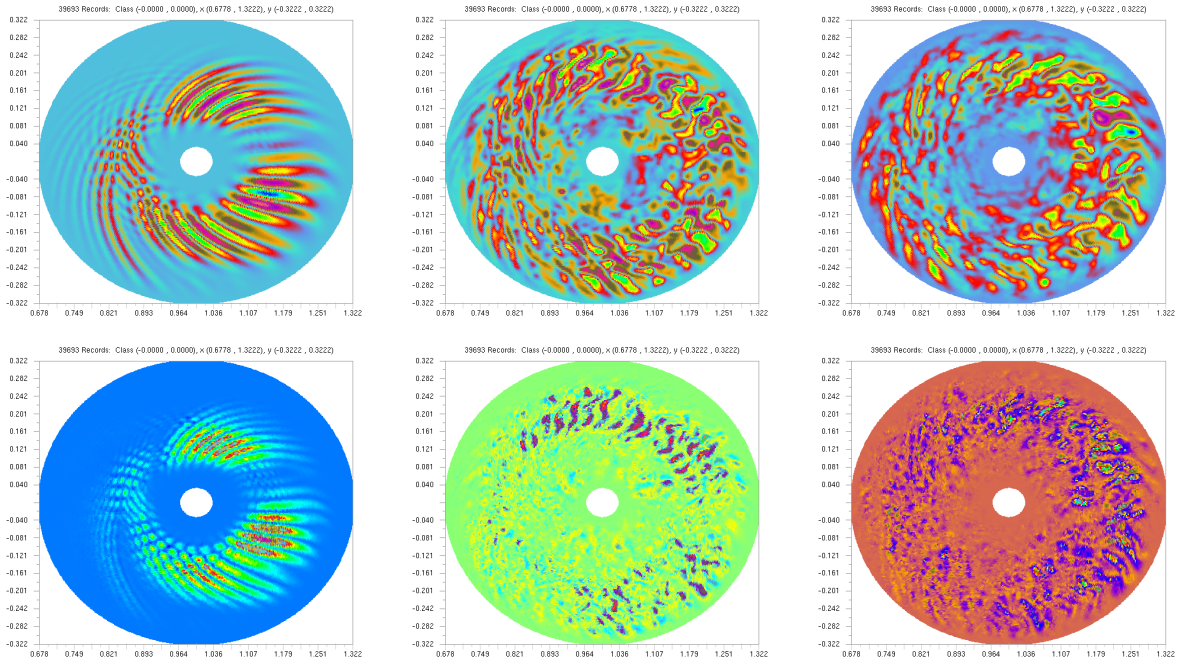

Figure 1: The electrostatic potential (top) and the ion heat flux (bottom) variables at time steps 1500 (left), 2500 (middle), and 3500 (right) in a plasma physics simulation. Points with similar colors have similar values, but the colors themselves do not have any significance.

The images were generated using a tool that assigns a color to each grid point based on the value of the variable. Points with similar colors have similar values, but the colors themselves do not have any significance. Note that the structures in both the variables are well formed at time step 1500, but degenerate as time progresses. Figure 2 shows a zoomed-in of the two variables, clearly showing the grid points. The goal of the analysis is to understand turbulence in plasma by identifying the structures, extracting statistics on them, and tracking them over time.

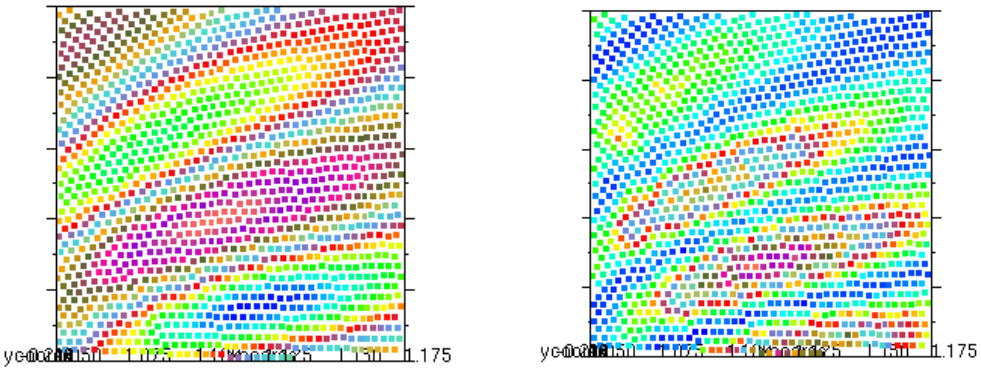

Figure 2: A zoomed-in view of the electrostatic potential (left) and the ion heat flux (right) variables at time steps 1500. Note that the ion heat flux is noisier than the electrostatic potential as indicated by the different colored points that form the interior of each structure.

## 3 Extending a threshold-based algorithm to extreme-scale

In earlier work [8], we used a simple threshold-based algorithm to separate the coherent structures in the ion heat flux (the variable of interest) from the background. This solution was motivated by the fact that we could exploit the values of the ion heat flux on a flux surface (the set of grid points at a fixed radius from the center of the poloidal plane), as shown in Figure 3, to calculate a threshold that would disconnect the structures from each other. At later time steps, when the ion heat flux is noisier, we can exploit the better-behaved electrostatic potential to obtain the threshold.

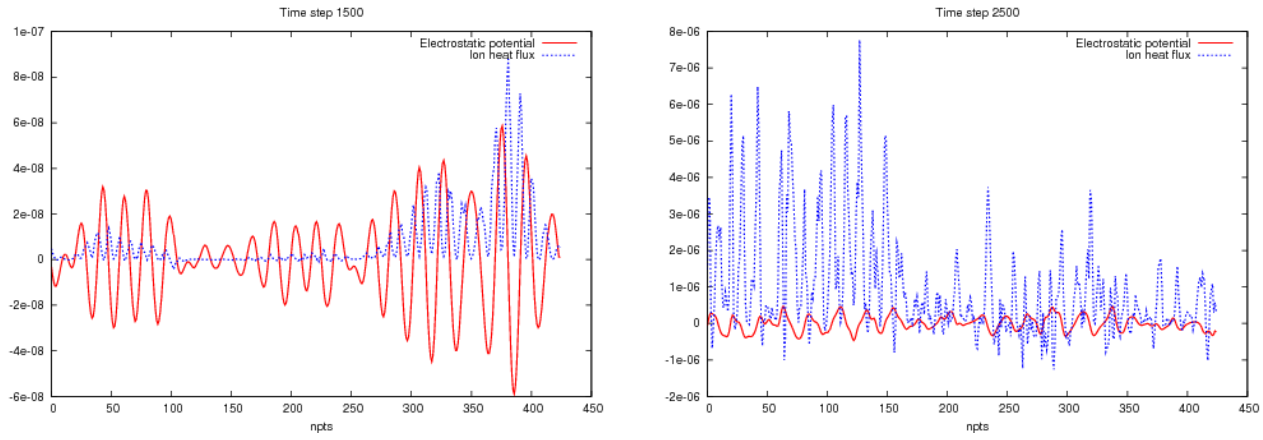


Figure 3: The electrostatic potential (red) and ion heat flux (blue) values on a flux surface at time steps 1500 (left) and 2500 (right). Observe in the left figure that a threshold selected at the valleys of the ion heat flux will disconnect neighboring structures in this variable.

A serial algorithm for finding the structures in this dataset can be summarized as follows:

- **Step 1:** Determine a threshold for a poloidal plane by considering the values of the ion heat flux and the electrostatic potential at a few flux surfaces and taking the average of the thresholds obtained for each flux surface. Next, obtain the global threshold for a time step by averaging the values of the thresholds for each poloidal plane.
- **Step 2:** Apply the threshold to disconnect the structures from each other. A simple approach that drops all grid points whose value is below the threshold can result in structures which are connected to each other at a single point, resulting in artificially large structures. To avoid this, we use additional constraints, such as "a grid point is retained as part of a structure, if it, and $n$ of its $m$ neighbors have values higher than the threshold." Such constraints address points at the boundary of a structure and disconnect structures that would have been connected if a simple thresholding had been used.
- **Step 3:** Use a connected component analysis [1] based on a union-find data structure to associate a unique identifier with all grid points corresponding to a coherent structure.

Once we have identified the grid points that form a structure (see Figure 4), we can extract statistics such as the sizes of the structures, the distribution of the sizes, the distribution of the integrated ion heat flux in the structures, and so on.

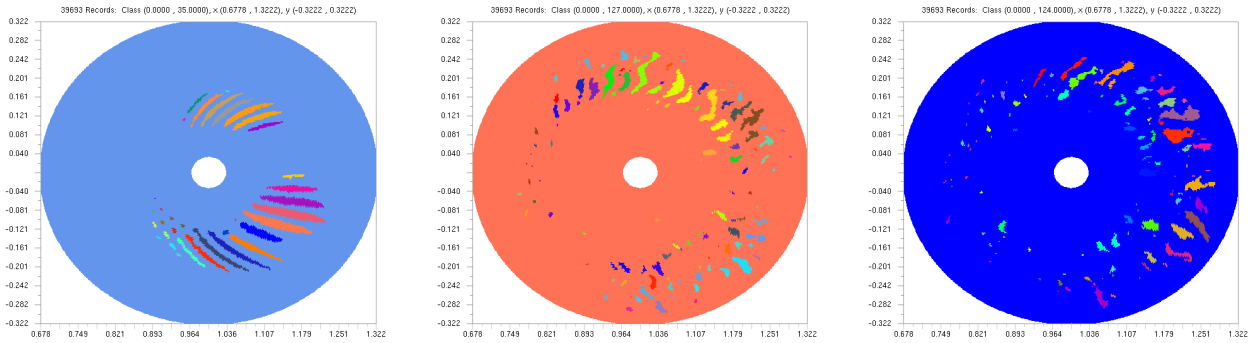

Figure 4: The coherent structures identified in the ion heat flux variable at time steps 1500 (left), 2500 (middle), and 3500 (right). Points within a structure are assigned the same color.

For the parallel, *in situ* implementation, we assume a general case, where each processor has grid points that correspond to parts of several flux surfaces. The calculation of the threshold (Step 1) is relatively simple as it requires minimal additional memory and communication. The more challenging aspects of parallelizing this algorithm, without increasing the memory requirements or communication costs substantially, lie in the application of the threshold (Step 2) and the connected component analysis for the identification of the grid points corresponding to each structure (Step 3). We next discuss our work on the first of these problems.

The application of the threshold requires the identification of the neighbors of each grid point. Since the grid is fixed for all time steps, the current aproach calculates the neighbors of each grid point on each poloidal plane, once, offline, and writes out the information to a file. The neighbors of a point are defined as grid points within a certain fixed radius of the point. This file is read in for each poloidal plane at each time step for use in the coherent structure analysis, an approach which will be infeasible in extreme scale.

We are exploring several options for calculating the nearest neighbors, an operation which occurs in many algorithms which are used for coherent structure detection as these structures are identified based on the similarity of neighboring points. We consider three options (i) a brute force search, where no additional memory is required; (ii) a simple uniform grid [2] that has low memory requirements and has shown to be scalable to large number of points in two- and three-dimensions; and (iii) a more complex data structure, such as the KD tree [1], which is re-created at each time step to speed up the search while avoiding a permanent increase in the memory by keeping the structure across time steps.

Figure 5 shows that the time it takes to conduct a nearest neighbor search varies for these three methods. As expected, the uniform grid works well at low dimensions ($< 5$), but its performance rapidly worsens with increasing dimension. A similar behavior is seen with the KD-tree data structure, though it performs well for dimension less than 18, and is better than a brute force search until dimension 25. However, there is a trade-off between the time required for search and the time it requires to build the data structure, as well as the additional memory required to store the data structure. Since we need to perform the nearest neighbor search in two dimensions, we expect that the uniform grid will perform better than the KD tree. We also expect that, given a dataset, each method will have its advantages and disadvantages, and the optimal choice of a data structure (if any) will depend on the relative "cost" of the additional memory required, the data movement involved in the brute force search, and the size of the data on each processor. As we implement our algorithms in parallel, we continue to explore these alternatives, as well as approaches that can exploit any additional information specific to the dataset to reduce the search time and the memory required.

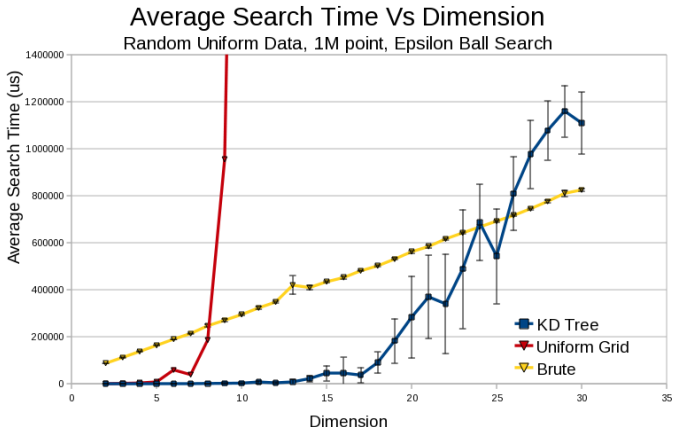


Figure 5: The average search time as a function of the dimension of the data using a brute force search, a uniform grid, and a KD-Tree data structure.

# 4 Fast algorithms for graph-based compression

In our work, we are also exploring graph-based techniques to compress the data that are being output by the simulation. Specifically, we are investigating the effectiveness of a class of lossy compression approaches that replace the actual values associated with sets of grid-nodes with a constant value whose difference from the actual value is bounded by a user-supplied error tolerance parameter. More details on our work is available in [5]; we provide a brief summary below.

We model the grid underlying the simulation data via a *graph* $G = (V, E, L)$. The set of vertices $V$, models the nodes of the grid for which values are computed. The set of edges $E$, models the connectivity of adjacent nodes. Two nodes are adjacent if they belong to the same element in the grid. The set of vertex-labels $L$, models the values computed at each node of the grid such that $l_i$ stores the value computed for node $v_i$. In this initial work, we assume there is only one value being computed for each node of the grid.

An *$\epsilon$-bounded set-based decomposition* of $G$ is a partitioning of its set of vertices into non-overlapping sets $\{V_1, \ldots, V_k\}$ such that for each $V_i$, $\forall v_q, v_r \in V_i$, $|l_q - l_r| \leq \epsilon$ (i.e., each set contains vertices whose values differ at most by $\epsilon$). When the induced subgraph $R_i = (V_i, E_i)$ of $G$ is connected, the set $V_i$ will also be referred to as a *region* of $G$. When all sets in an $\epsilon$-bounded set-based decomposition form regions, then the decomposition will be referred to as an *$\epsilon$-bounded region-based decomposition* of $G$. Given a set of vertices $V_i$, the average value of its vertices will be referred to as its *mean value* and will be denoted by $\mu(V_i)$.

We developed two classes of approaches for obtaining the $\epsilon$-bounded set-based decomposition of $G$. The first class, referred to as the *set-based decomposition* focuses entirely on the vertices of the grid and their values, while the second class, the *region-based decomposition* also takes into account the connectivity of these

| Dataset | $\|V\|$ | $\|E\|$ | $\mu(V)$ | Grid Type |
|---|---|---|---|---|
| d1 | 486051 | 4335611 | 0.9958 | unstruct. |
| d2 | 589824 | 1744896 | 0.5430 | struct. |
| d3 | 1936470 | 15399496 | 0.9874 | unstruct. |
| d4 | 16777216 | 50102272 | 163.70 | struct. |

Table 1: Datasets used in exploring compression techniques.

vertices in the graph. In addition, we developed different approaches for encoding the information that needs to be stored on the disk in order to maximize the overall compression. In developing these approaches, our research focused on algorithms whose underlying computational complexity is low because we are interested in being able to perform the compression *in-situ* with the execution of the scientific simulation on future exascale-class parallel systems. As a result of this design choice, the algorithms that we present tend to find sub-optimal solutions but do so in time that in most cases is bounded by $O(|V| \log |V| + |E|)$.

We evaluated our algorithms using seven real world datasets in fluid turbulence and combustion simulations with both structured and unstructured grids. We present results for the subset of four datasets described in Table 4. We measured the performance of the different compression approaches using both the error introduced by the lossy compression and the degree of compression that was achieved. The error was measured using three different metrics: (i) the root mean squared error (RMSE), defined as:

$$RMSE = \sqrt{\frac{1}{|V|} \sum_{i=1}^{|V|} |l_j - \hat{l}_j|^2}, \tag{1}$$

where $l_j$ is the original value of vertex $v_j$ and $\hat{l}_j$ is its reconstructed value; (ii) the maximum point-wise error (MPE), defined as:

$$MPE = \max(|l_1 - \hat{l}_1|, ..., |l_n - \hat{l}_n|), \tag{2}$$

which is the $\ell_\infty$-norm of the point-wise error vector; and (iii) the peak signal-to-noise ratio (PSNR), defined as:

$$PSNR = 20 \cdot \log_{10} \left( \frac{\max(x_1, ..., x_n)}{RMSE} \right), \tag{3}$$

which is a normalized error measure, thus facilitating comparisons of error between datasets with values that differ greatly in magnitude. The MPE measure is presented in tandem with RMSE to identify those algorithms which achieve low RMSE, but sustain high point-wise errors.

The compression effectiveness was measured by computing the compression ratio (CR) of each method, defined as:

$$CR = \frac{\text{compressed size}}{\text{uncompressed size}}. \tag{4}$$

To evaluate our methods, we first selected a fixed set of values for RMSE and compared the various algorithmic choices for the set- and region-based decomposition approaches in terms of their compression ability. Then, we compared the compression performance of the best combinations of these schemes against that achieved by other approaches on two levels of lossy compression errors. We considered three other popular compression schemes: the wavelet method [12, 13] which is often used for visualization in the context of structured grids; spectral compression [9] that extends the discrete cosine transform used in JPEG from 2D regular grids to the space of any dimensional unstructured grids; and Adaptive Coarsening (AC) [15] which is based on down-sampling a mesh in areas which can be reconstructed within some error tolerance and storing at full resolution the others.

Table 4 presents the results for our best methods (indicated by SBD1 and RBD2) against the wavelet compression (Wvlt), spectral compression (Spctrl), and adaptive coarsening (AC) methods using a high level of compression error on the four datasets in Table 4.

Additional details on all our experiments, including those on other datasets and at other levels of tolerance are provided in [5]. We next summarize our results on all the experiments we performed, not just those presented in this paper. We found that, on average, our algorithms compress the simulation datasets to

| info | | high error tolerance | | | |
|---|---|---|---|---|---|
| Dataset | Algorithm | RMSE | PSNR | MPE | CR |
| | SBD1 | 6.30E-03 | 4.64E+01 | 1.89E-02 | 2.39E-02 |
| d1 | RBD2 | 6.28E-03 | 4.65E+01 | 1.89E-02 | **2.52E-02** |
| | Spctrl | 6.37E-03 | 4.63E+01 | 1.11E-01 | 4.00E-02 |
| | SBD1 | 2.92E-02 | 3.60E+01 | 7.33E-02 | **2.51E-03** |
| | RBD2 | 2.88E-02 | 3.61E+01 | 8.02E-02 | 5.02E-03 |
| d2 | Wvlt | 3.10E-02 | 3.55E+01 | 2.34E-01 | 2.00E-02 |
| | Spctrl | 3.17E-02 | 3.53E+01 | 7.34E-01 | 4.50E-02 |
| | AC | 3.31E-02 | 3.49E+01 | 1.50E-01 | 1.86E-02 |
| | SBD1 | 5.22E-03 | 4.88E+01 | 1.91E-02 | **1.27E-02** |
| d3 | RBD2 | 5.18E-03 | 4.89E+01 | 1.93E-02 | 1.33E-02 |
| | Spctrl | 5.27E-03 | 4.87E+01 | 2.14E-01 | 4.50E-02 |
| | SBD1 | 2.36E+01 | 4.70E+01 | 1.63E+02 | **2.63E-03** |
| | RBD2 | 2.05E+01 | 4.83E+01 | 1.65E+02 | 6.30E-03 |
| d4 | Wvlt | 2.47E+01 | 4.66E+01 | 6.86E+02 | 7.50E-03 |
| | Spctrl | 2.57E+01 | 4.63E+01 | 1.78E+03 | 3.50E-02 |
| | AC | 2.30E+01 | 4.73E+01 | 3.01E+03 | 2.15E-02 |

bold indicates the lowest CR for a given dataset and error tolerance

Table 2: Comparison of scientific data compression algorithms for the high error tolerance case.

2–5% of their original size. Compared with just lossless compression only, which results in storage costs of 40–80% of the original size, this is a big improvement. The results also show that for all but two experiments, SBD1 performs the best and that on average it required only 36% of the storage of the next best algorithm. For unstructured grids it requires on average 25% of the storage of Spctrl whereas for structured grids it requires on average 48% and 38% of the space of Wvlt and AC, respectively. Moreover, we see that as the amount of allowable error is lowered, the performance gap between SBD1 and the other methods grows. In addition, for unstructured grids, RBD2 performs the second best overall and requiring 61% of the space required by the Spctrl on average. Over our benchmark suite, our methods obtained compression of 1% of the original size with average PSNR of 43.00 and 3% of the original size with average PSNR of 63.30. Our experiments show that our methods achieve compressed representations, which on average, require 50%–75% less space than competing schemes at similar or lower reconstruction errors. These results suggest that in the context of grid-based simulation, SBD1 and RBD2 are consistently good choices for compression, providing low point-wise and global reconstruction error, high compression ratio, and low computational complexity.

# 5 Summary and future work

In this paper, we described the challenges encountered in the analysis of coherent structures, a common task in three-dimensional simulations of phenomenon in many scientific domains. As we move towards future extreme scale systems, we expect that the I/O bandwidth will no longer support the writing out of data for analysis off-line. Moving the analysis *in situ* is an option only in some cases, but we still need to adapt to the relatively smaller memory sizes and the high cost of data movement. To address this, we are exploring ways in which we can modify an existing algorithm for coherent structure analysis so it can co-exist with the simulation on an extreme-scale system. We are also considering compression techniques to intelligently reduce the size of the data output. This would enable us to use the reduced data for analysis and to address questions that might not have been formulated when the simulation was run, thus supporting scientific discovery at the exascale.

The next steps in our research will focus on parallel implementations of the work presented in this paper, with an emphasis on reducing data movement and memory requirements.

# 6 Acknowledgment

We would like to thank our domain collaborators for graciously sharing their datasets with us. This work was supported in part by the DOE Office of Science, Advanced Scientific Computing Research, program manager, Lucy Nowell. The work on nearest neighbor searches was performed by Roger Seth Kirk while he was at Lawrence Livermore National Laboratory as a summer intern from the University of Kentucky.

LLNL-CONF-541611 The work of Chandrika Kamath and Roger Kirk was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

# References

[1] Cormen, T. H., Leiserson, C. E., and Stein, R. L. *Introduction to Algorithms*, third ed. MIT Press and McGraw-Hill, Cambridge, MA, 2009.

[2] Franklin, W. R. Nearest point query on 184,088,599 points in $E^3$ with a uniform grid. website, 2006. See: `http://www.ecse.rpi.edu/Homepages/wrf/pmwiki/pmwiki.php/Research/Nearpt3`.

[3] GSEP: Center for Gyrokinetic Simulation of Energetic Particle Turbulence and Transport Web page, 2012. `http://phoenix.ps.uci.edu/gsep/`.

[4] ITER Web page, 2012. `http://www.iter.org/`.

[5] Iverson, J., Kamath, C., and Karypis, G. Fast and effective lossy compression algorithms for scientific datasets. Submitted for publication.

[6] Kamath, C., Gezahegne, A., and Miller, P. L. Identification of coherent structures in three-dimensional simulations of a fluid-mix problem. *International Journal of Image and Graphics 9* (2009), 389–410.

[7] Kamath, C., and Miller, P. L. Image analysis for validation of simulations of a fluid-mix problem. In *IEEE International Conference on Image Processing, Volume III* (September 2007), pp. 525–528.

[8] Kamath, C., Xiao, Y., and Lin, Z. Analysis of structures and event size statistics in plasma turbulence: Preliminary results. International Sherwood Fusion Theory Conference, Seattle, WA, April, 2010. See: `http://ckamath.org/publications_by_project/gsep_data_analysis`.

[9] Karni, Z., and Gotsman, C. Spectral compression of mesh geometry. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (2000), pp. 279–286.

[10] Lin, Z., Hahm, T. S., Lee, W. W., Tang, W. M., and White, R. B. Turbulent transport reduction by zonal flows: Massively parallel simulations. *Science 281* (1998), 1835–1837.

[11] Love, N. S., and Kamath, C. Image analysis for the identification of coherent structures in plasma. In *Proceedings, Applications of Digital Image Processing, XXX, SPIE Conference 6696* (2007), p. 66960D.

[12] Mallat, S. G. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 11* (1989), 674–693.

[13] Muraki, S. Volume data and wavelet transforms. *IEEE Computer Graphics and Applications 13*, 4 (1993), 50–56.

[14] Streitz, F. Keeping an eye on the prize. Science and Technology Magazine, Lawrence Livermore National Laboratory, 2006. See: `https://www.llnl.gov/str/JulAug06/Streitz.html`.

[15] Unat, D., Hromadka, T., and Baden, S. B. An adaptive sub-sampling method for in-memory compression of scientific data. In *Proceedings, Data Compression Conference* (2009), pp. 262–271.